Dafydd Jenkins, Josefina Jahde, Valentino Cannestraci, Luz Alba Posse.

# Trabajo Práctico 3 | TD II

**1. Introducción - Leer la hoja de datos y responder:**

a) ¿Cuál es el tamaño de la memoria en cantidad de bytes?

Sabiendo que la memoria consta de 256 palabras y teniendo como dato que una palabra tiene un tamaño de 8 bits, el tamaño de memoria en cantidad de bytes es de 256.

b) ¿Cuántas instrucciones sin operandos se podrían agregar al formato de instrucción?

Sabiendo que el opcode ocupa 5 bits hay 2^5 posibles operaciones indicables

c) ¿Qué tamaño tiene el PC?

El PC cuenta con un tamaño de 8 bits (1 byte).

d) ¿Dónde se encuentra y qué tamaño tiene el IR?

El IR se encuentra dentro del  decoder. Al saber que entran 8 bits por high y 8 por low, determinamos que cuenta con tamaño de 2 bytes.

e) ¿Cuál es el tamaño de la memoria de microinstrucciones? ¿Cuál es su unidad direccionable?

Esta memoria contiene palabras de 32 bits (unidad direccionable) y direcciones de 9 bits, lo cual implica una memoria de 2^9 posibles direcciones. Si cada dirección es de 32 bits, la memoria es de 2 KB direccionable a 4 bytes.

**2.  Analizar - Estudiar el funcionamiento de los circuitos indicados y responder las siguientes preguntas:**

a. PC (Contador de Programa): ¿Que función cumple la señal inc?

La función de la señal INC es incrementar el valor almacenado en el PC (Contador de Programa). Esta señal está directamente conectada a la unidad de control, lo que resulta en el aumento del valor actual del registro seleccionado. Este incremento permite realizar la búsqueda (fetch) de la instrucción siguiente en la secuencia de ejecución del programa.

b. ALU (Unidad Aritmético Lógica): ¿Qué función cumple la señal op_W?

La señal op_W cumple la función de habilitar la escritura de los flip flops que almacenan el valor de los flags de la ALU.

c. Control Unit (Unidad de control): ¿Cómo se resuelven los saltos condicionales? Describir detalladamente el mecanismo, incluyendo la forma en que interactúan las señales jc microOp, jz microOp, jn microOp y jo microOp, con los flags

   Los saltos condicionales se resuelven en la UC. Cuando en el programa se llama a las instrucciones, se evalúa con las compuertas AND si el flag a evaluar está encendido o no. Si el mismo lo está entonces el AND devuelve un 1 y se activa la microinstrucción que genera el salto.

d. microOrgaSmall (DataPath): ¿Para qué sirve la señal DE enOutImm? ¿Qué parte del circuito indica cuál índice del registro a leer y escribir?

   Lo que hace la señal DE_enOutImm es habilitar el componente de tres estados que habilita la salida del Decoder al BUS de los datos inmediatos. Las señales que salen de la UC (RB_selectIndexIn y RB_selectIndexOut), indican que registro leer y escribir. (0 si es RX y 1 si es RY).

**3. Ensamblar y ejecutar - Escribir el siguiente archivo, compilarlo y cargarlo en la memoria de la máquina:**

a) Previamente a ejecutar el programa, describir con palabras el comportamiento esperado del mismo. No se debe explicar instrucción por instrucción, la idea es entender que hace el programa y qué resultado genera.

El programa ejecuta un loop que llama a la subrutina *"shift"*, desplaza bits hacia la izquierda en el registro R2 y guarda ese valor en memoria a partir de la posición apuntada por R3. Luego se restaura el valor de R2 antes de ser duplicado y se le resta |1. El loop continúa hasta que el valor de R1 sea igual al resultado de restar R0 a R2 (hasta que R2 valga 0), momento en el cual el programa finaliza. En el ejemplo, como R2 comienza valiendo 16, el loop se ejecuta 16 veces, guardando 16 valores en memoria, cada uno en una posición entre 0x30 y 0x40.

b) Identificar la dirección de memoria de cada una de las etiquetas del programa

start: 0x00
loop: 0x0A
shift: 0x16
end: 0x14

c) Ejecutar e identificar cuántos ciclos de clock son necesarios para que el programa llegue a la instrucción JMP halt.

Para resolver este ejercicio construimos un pequeño circuito conectado al PC y al clock que cuente ciclos de clock hasta que el PC llegue a 15 (cuando pasa 14, dirección de la etiqueta end). Para que se llegue a ejecutar (para que el programa entre al loop infinito final por primera vez) son necesarios 1515 ciclos de clock. Para que el PC llegue a 0x14 (dirección de la etiqueta end) por primera vez, son necesarios 121.

d) ¿Cuántas microinstrucciones son necesarias para ejecutar la instrucción ADD? ¿Cuántas para la instrucción JZ? ¿Cuántas para la instrucción JMP?

*ADD*:

- Cargar 1er operando
- Cargar 2do operando
- Sumar
- Guarda el resultado

Se necesitan 4 microinstrucciones.

*JZ*:

- Cargar 1er operando
- Cargar 2do operando
- Comparar entre 2 valores
- Se evalúa si el resultado de la comparación da 0 o 1, para determinar si se cumple la igualdad
- Si la condición es verdadera, se salta a la dirección especificada. O sea, cargando la dirección de distinto del salto al PC

Se necesitan 5 microinstrucciones.

*JMP*:

- Cargar la dirección del salto
- Saltar

Se necesitan 2 microinstrucciones.

**e) Describir detalladamente el funcionamiento de las instrucciones PUSH, POP, CALL y RET.**

*PUSH*: la instrucción PUSH cumple la función de agregar su operando a la última posición del stack. En otras palabras, guarda el contenido del registro pasado por parámetro en el stack.

*POP*: Saca el último elemento del stack, y guarda el contenido de ese elemento en el registro pasado por parámetro.

*CALL*: Llama a subrutinas o función. Salta a la dirección especificada (modifica el PC), guardando en el registro pasado por parámetro (normalmente el stack) el PC actual.

*RET*: Retorna la subrutina y continúa con la ejecución del programa principal. Establece el PC en el valor pasado por parámetro (valor normalmente guardado en el stack).

**4. Programar - Escribir en ASM las siguientes funciones:**

La respuesta está en los archivos.